Fusing Spatial, Pictorial and Photometric Data to Build Photorealistic Models

Z. Jankó[*] MTA SZTAKI, Budapest

E. Lomonosov[†] MTA SZTAKI, Budapest

D. Chetverikov[‡] MTA SZTAKI, Budapest

Abstract

We are working on several projects related to the automatic fusion and high-level interpretation of 3D sensor data for building models of real-world objects and scenes. Our major goal is to create rich and geometrically correct, scalable photorealistic 3D models based on multimodal data obtained using a laser scanner, a camera and illumination sources. In this report, we present a sophisticated software system that processes and fuses geometric, pictorial and photometric data using genetic algorithms and efficient methods of computer vision.

Keywords: photorealistic models, data fusion, photometric stereo, genetic algorithms

1 Introduction

Building photorealistic 3D models of real-world objects is a fundamental problem in computer vision and computer graphics. Such models require precise geometry as well as detailed texture on the surface. Textures allow one to obtain visual effects that are essential for high-quality rendering. Photorealism is further enhanced by adding surface roughness in form of the so-called 3D texture represented by a bump map.

Different techniques exist to reconstruct the object surface and to build photorealistic 3D models. Although the geometry can be measured by various methods of computer vision, for precise measurements laser scanners are usually used. However, most of laser scanners do not provide texture and colour information, or if they do, the data is not accurate enough. (See [Yemez and Schmitt 2004] for a detailed discussion.)

Our primary goal is to create a system that only uses a PC, an affordable laser scanner and a commercial (although high-quality) uncalibrated digital camera. The camera should be used freely and independently from the scanner. No other equipment (special illumination, calibrated setup, etc.) should be used. No specially trained personnel should be needed to operate the system: After training, a computer user with minimal engineering skills should be able to use it. The ambitious projects [Bernardini 2002; Ikeuchi 2003; M. Levoy 2000] have developed sophisticated technologies for digitising statues and even buildings, but these technologies are extremely expensive and time-consuming due to the size of the objects to be measured. They require specially designed equipment and trained personnel. Creation of a model takes weeks [Bernardini 2002] or even months.

Our modelling system receives as input two datasets of diverse origin: a number of partial measurements (3D point sets) of the object surface made by a hand-held laser scanner, and a collection of high quality images of the object acquired independently by a digital camera using a number of illumination sources. The partial surface measurements overlap and cover the entire surface of the object; however, their relative orientations are unknown since they are obtained in different, unregistered coordinate systems. A specially designed genetic algorithm (GA) automatically pre-aligns the surfaces and estimates their overlap. Then a precise and robust iterative algorithm (Trimmed Iterative Closest Point, TrICP [Chetverikov et al. 2005]) developed in our lab is applied to the roughly aligned surfaces to obtain a precise registration. Finally, a complete geometric model is created by triangulating the integrated point set.

The geometric model is precise, but it lacks texture and colour information. The latter is provided by the other dataset, the collection of digital images. The task of precise fusion of the geometric and the visual data is not trivial, since the pictures are taken freely from different viewpoints and with varying zoom. The data fusion problem is formulated as photo-consistency optimisation, which amounts to minimising a cost function with numerous variables which are the internal and the external parameters of the camera. Another dedicated genetic algorithm is used to minimise this cost function.

When the image-to-surface registration problem is solved, we still face the problem of seamless blending of multiple textures, that is, images of a surface patch appearing in different views. This problem is solved by a surface flattening surface algorithm that gives a 2D parametrisation of the model. Using a measure of visibility as weight, we blend the textures providing a seamless and detail-preserving solution. Finally, photometric data is added to provide a bump map reflecting the surface roughness.

All major components of the described system are original, developed in our laboratory. Below, we present the main algorithms and give examples of photorealistic model building using GA-based registration and fusion of spatial, pictorial data and photometric data. Most of this report is a short version of the book chapter [Chetverikov et al. 2006] that describes our system in full detail and provides numerous test data. The section that presents our initial results with the photometric data is new.

2 Pre-registration of surfaces using a genetic algorithm

This section deals with genetic pre-alignment of two arbitrarily oriented datasets, which are partial surface measurements of the object whose model we wish to build. (See figure 1 for an illustration of such measurements.) The task is to quickly obtain a rough pre-alignment suitable for subsequent application of the robust Trimmed Iterative Closest Point algorithm [Chetverikov et al. 2005] developed in our lab earlier.

Consider two partially overlapping 3D point sets, the *data* set $\mathscr{P} = \{\mathbf{p}_i\}_1^{N_p}$ and the *model* set $\mathscr{M} = \{\mathbf{m}_i\}_1^{N_m}$. Denote the overlap by ξ. Then the number of points in $\mathscr{P}$ that have a corresponding point in $\mathscr{M}$ is $N_{po} = \lfloor \xi N_p \rfloor$. The standard ICP [Besl and McKay 1992] assumes that $\mathscr{P}$ is a subset of $\mathscr{M}$. ICP iteratively moves $\mathscr{P}$ onto $\mathscr{M}$ while pairing each point of $\mathscr{P}$ with the closest point of $\mathscr{M}$. The

[*] e-mail: janko@sztaki.hu

[†] e-mail: zhenya@tinos.geo.sztaki.hu

[‡] e-mail: csetverikov@sztaki.hu

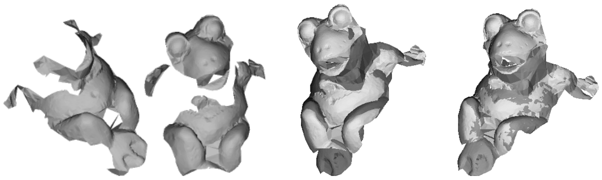


Figure 1: The Frog dataset, GA alignment and final alignment.

cost function of ICP is the mean square error (MSE), that is, the mean of all residuals (distances between paired points).

In contrast to ICP, our TrICP [Chetverikov et al. 2005] only assumes a partial overlap of the two sets, which is more realistic. TrICP finds the Euclidean motion that brings an N_{po}-point subset of $\mathscr{P}$ into the best possible alignment with $\mathscr{M}$. The algorithm uses another cost function. At each iteration, N_{po} points with the least residuals are selected, and the optimal motion is calculated for this subset so as to minimise the trimmed MSE

$$e = \frac{1}{N_{po}} \sum_{i=1}^{N_{po}} d^2_{i:N_p}, \tag{1}$$

where $\{d^2_{i:N_p}\}_1^{N_p}$ are the sorted residuals. The subset of the N_{po} paired points is iteratively updated after each motion.

In practice, the overlap ξ is usually unknown. It can be set automatically [Chetverikov et al. 2005] by running TrICP for different values of ξ and finding the minimum of the objective function

$$\Psi(\xi, \mathbf{R}, \mathbf{t}) = \frac{e(\xi, \mathbf{R}, \mathbf{t})}{\xi^2}, \tag{2}$$

which minimises the trimmed MSE while trying to use as many points as possible.

When an object is scanned by a 3D scanner, $\mathscr{P}$ and $\mathscr{M}$ are often obtained in different coordinate systems. As a result, their orientations may be very different. TrICP provides an efficient and robust solution when the two sets are roughly pre-registered. This is typical for all iterative algorithms, for which the pre-alignment is usually done manually. Our genetic pre-registration procedure [Lomonosov et al. 2006] complements TrICP yielding a robust and completely automatic solution.

The genetic pre-registration algorithm minimises the same objective function $\Psi(\xi, \mathbf{R}, \mathbf{t})$ as TrICP, but this time as a function of all the seven parameters, namely, the overlap ξ, the three components of the translation vector $\mathbf{t}$, and the three Euler angles of the rotation matrix $\mathbf{R}$. The difference between the genetic solution and the overlap selection procedure [Chetverikov et al. 2005] is essential. The former means evaluating $\Psi(\xi, \mathbf{R}, \mathbf{t})$ for different values of ξ, $\mathbf{R}$, and $\mathbf{t}$, while the latter means running TrICP for different values of ξ. Our genetic solution provides an elegant way to estimate the overlap *and* the optimal motion simultaneously, by treating all parameters in a uniform way. The solution [Chetverikov et al. 2005] only works for pre-registered sets. If desired, it can be used to refine the overlap estimate obtained by the GA.

To minimise the objective function $\Psi(\xi, \mathbf{R}, \mathbf{t})$, we applied a genetic algorithm tuned to the problem. The objective function was evaluated by mapping each integer parameter onto a real-valued range using normalisation. Simple one-point crossover was employed. Different population sizes were tested and an optimal value was selected for the final experiments. Two mutation operators were introduced. Shift mutation shifts one parameter randomly by a value not exceeding 10% of the parameter range, while replacement mutation replaces a parameter with a random value. The corresponding probabilities were also set after preliminary experimentation. Tournament selection was applied, as it is easy to implement and helps avoid premature convergence. An elitist genetic algorithm was employed, where one copy of the best individual was transferred without change from each generation to the next one. The method is presented in detail in our paper [Lomonosov et al. 2006].

We have tested the genetic pre-alignment and the combined method (GA followed by TrICP) on different data. To test the method under arbitrary initial orientations, set $\mathscr{P}$ was randomly rotated prior to alignment in each of the 100 tests. Results of all tests were visually checked. No erroneous registration was observed. Typical results of alignment are displayed in figures 1, 3 and 2. In each figure, the first two pictures show the two datasets to be registered. The datasets result from two separate measurements of the same object obtained from different angles.

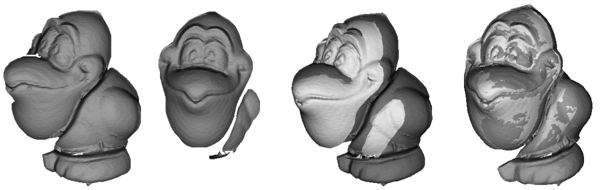


Figure 2: The Bird dataset, GA alignment and final alignment.

The third picture of each figure (GA) displays the result of our genetic pre-registration algorithm. Here, the two datasets are shown in different colours. One can see that the datasets are roughly registered, but the registration quality is not high: the surfaces are displaced, and they occlude each other in large continuous areas instead of ‘interweaving’. Finally, the rightmost picture is the result of the fine registration obtained by TrICP using the result of the genetic pre-registration. Here, the surfaces match much better, and they are interwoven, which is an indication of the good quality of the final registration.

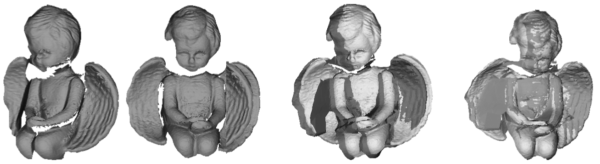


Figure 3: The Angel dataset, GA alignment and final alignment.

3 Fusion of surface and image data

In this section, we address the problem of combining geometric and textural information of the object. As already mentioned, the two sources are independent in our system: the 3D geometric model is obtained by 3D scanner, then covered by high quality optical images. After a brief survey of relevant previous work, we discuss our photo-consistency based registration method with genetic algorithm based optimisation. Then we deal with the task of blending multiple texture mappings and present a novel method which combines the techniques of surface flattening and texture merging. Finally, initial results on using the photometric data to add surface roughness are shown.

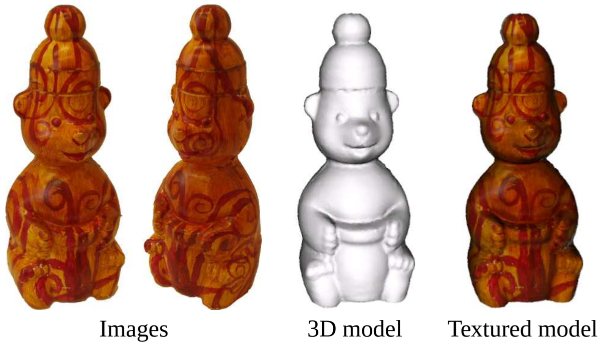


Figure 4: The Bear Dataset and result of registration of images to surface.

3.1 Registering images to a surface model

Several 2D-to-3D (image-to-surface) registration methods were proposed in computer vision and its medical applications. Most of them are based on feature correspondence: Feature points are extracted both on the 3D surface and in the images, and correspondences are searched for. (See, for example, [David 2002; Haralick 1989; Leventon et al. 1997].) However, the features are often difficult to localise precisely in 3D models. In addition, defining similarity between 2D and 3D features is not easy.

Intensity based registration is another approach to the problem. The algorithm of Clarkson et al. [Clarkson 2001] applies the photo-consistency to find the precise registration of 2D optical images of a human face to a 3D surface model. They use calibrated images, thus the problem is reduced to estimating the pose of the cameras. We do not use calibrated camera, so the number of parameters is much higher. The size of the parameter space and the behaviour of the cost function motivated the use of genetic algorithm-based optimisation.

The input data consists of two colour images, I_1 and I_2, and a 3D surface model. They represent the same object. (See figure 4 for an example.) The images are acquired under fixed lighting conditions and with the same camera sensitivity. All other camera parameters may differ and are unknown. The raw 3D data is processed by the efficient and robust triangulator [Kós 2001] developed in our lab. The 3D model obtained consists of a triangulated 3D point set (mesh) $\mathscr{P}$ with normal vectors assigned.

The finite projective camera model is used to project the object surface to the image plane: $\mathbf{u} \simeq P\mathbf{X}$, where $\mathbf{u}$ is an image point, P the 3×4 projection matrix and $\mathbf{X}$ a surface point. ($\simeq$ means that the projection is defined up to an unknown scale.) The task of registration is to determine the precise projection matrices, P_1 and P_2, for both images. Since the projection matrix is up to a scale factor, it has only 11 degrees of freedom in spite of having 12 elements. The collection of the 11 unknown parameters is denoted by p, which represents the projection matrix P as an 11-dimensional parameter vector.

Values of p_1 and p_2 are sought such that the images are *consistent* in the sense that the corresponding points – different projections of the same 3D point – have the same colour value. Assuming Lambertian surfaces, the formal definition is the following: We say that images I_1 and I_2 are consistent by P_1 and P_2 (or p_1 and p_2) if for each $X \in \mathscr{P}$: $\mathbf{u}_1 = P_1\mathbf{X}$, $\mathbf{u}_2 = P_2\mathbf{X}$ and $I_1(\mathbf{u}_1) = I_2(\mathbf{u}_2)$. (Here $I_i(\mathbf{u}_i)$ is the colour value in point $\mathbf{u}_i$ of image I_i.) This type of consistency is called **photo-consistency** [Clarkson 2001; Kutulakos and Seitz 1993].

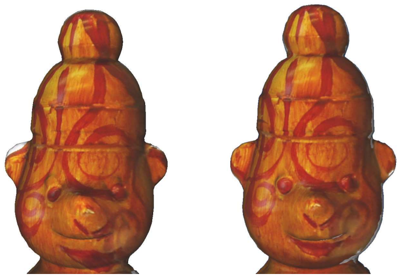


Figure 5: Difference between manual pre-registration and genetic registration.

The photo-consistency holds for accurate estimates for p_1 and p_2. Inversely, misregistered projection matrices mean much less photo-consistent images. The cost function is the following:

$$C_\phi(p_1, p_2) = \frac{1}{|\mathscr{P}|} \sum_{\mathbf{X} \in \mathscr{P}} \|I_1(P_1\mathbf{X}) - I_2(P_2\mathbf{X})\|^2 . \qquad (3)$$

Here ϕ stands for *photo-inconsistency* while $|\mathscr{P}|$ is the number of points in $\mathscr{P}$. Difference of the colour values $\|I_1 - I_2\|$ can be defined by a number of different colour models. (For details see [Jankó and Chetverikov 2004].) Finding the minimum of the cost function (3) over p_1 and p_2 yields estimates for the projection matrices.

We pre-register the images and the 3D model manually. This yields a good initial state for the search, which narrows the search domain and accelerates the method. Manual pre-registration is reasonable since this operation is simple and fast compared to the 3D scanning, which is also done manually. The photo-consistency based registration makes the result more accurate.

The genetic algorithm starts by creating the initial population. The individuals of the population are chosen from the neighbourhood of the parameter vector obtained by the manual pre-registration. The values of the genes are from the intervals defined by the pre-registered values plus a margin of $\pm\varepsilon$. In our experiments ε was set to values between 1% and 3%, depending on the meaning and the importance of the corresponding parameter. The individual that encodes the pre-registered parameter vector is also inserted in the initial population to avoid losing it.

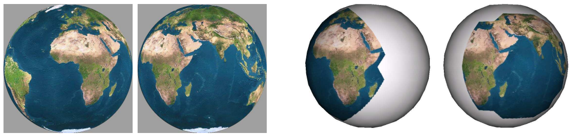
We applied the method to different real data. One of them, the Bear Dataset, is shown in figure 4. The precision of the registration can be best judged at the mouth, the eyes, the hand and the feet of the Bear. Figure 5 visualises the difference between the manual pre-registration and the photo-consistency based registration. The areas of the mouth, the eyes and the ears show the improvement of the quality.

3.2 Merging multiple textures

When the image-to-surface registration problem is solved, we still face the problem of seamless merging (blending) of multiple textures, that is, images of a surface patch appearing in different views. There are several ways to paste texture to the surface of an object. Graphics systems usually have the requirement of two pieces of information: a *texture map* and the *texture coordinates*. The texture map is the image we paste, while the texture coordinates specify where it is mapped to. Texture maps are usually two-dimensional, although during the last years the application of 3D textures has also become general.

It is straightforward to choose a photo of an object to be a texture map. An optical image of an object contains the colour information we need to paste to the surface. Projection of a 3D surface point $\mathbf{X}$ can be described in matrix form: $\widetilde{\mathbf{v}} \simeq P\widetilde{\mathbf{X}}$, where P is the 3×4 projection matrix and $\widetilde{}$ means homogenous coordinates [Hartley and Zisserman 2000]. In this way the texture mapping function is a simple projective transformation.

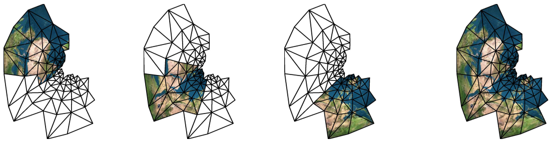
The difficulty of image-based texturing originates from the problem of occlusion, which yields uncovered areas on the surface. An optical image shows the object from only a single view. Therefore, it contains textural information only about the visible parts of the surface; the occluded areas are hidden from that view. (See example in figure 6b.) This insufficiency can be reduced – in optimal cases eliminated – by combining several images.



(a) Input images (b) Partially textured models

Figure 6: Textures cover only parts of the model.

Using the efficient flattening algorithm [Kós and Várady 2003] developed in our lab, we designed a flattening-based method to create a texture map based on optical images. Flattening the surface mesh of an object provides an alternative two-dimensional parametrisation. The advantage is that this parametrisation preserves the topology of the three-dimensional mesh. A texture that covers entirely the flattened 2D surface covers also the original 3D surface. Transforming optical images to flattened surfaces provides partial texture maps. (See figure 7.) But since flattening preserves the structure of the 3D mesh, these texture maps can be merged, in contrast to the original optical images.

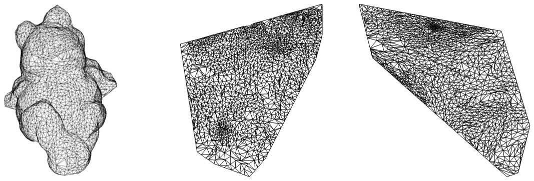


Partial textures Merged texture

Figure 7: Partial and merged texture maps.

Usually, complex meshes cannot be flattened at once, they need to be cut before flattening. We have chosen to cut by plane, since the cutting plane can be easily determined manually: three points selected on the surface define a plane. The 3D mesh is cut in half by this plane, then the halves are flattened and textured separately. The problem of re-merging the textured halves will be discussed later. Figure 8 shows an example of using the algorithm in our experiments.

After flattening the 3D surface, we convert optical images to flattened texture maps. In contrast to the projection matrix, this mapping is complicated, since neither the transformation of flattening nor the relation between the optical image and the texture map can be represented by a matrix. We use the mesh representation of the 3D surface for conversion: Given a triangle of the mesh, the vertices of the corresponding triangles are known both in the optical image and on the flattened surface. Let us denote these triangles by T_i in the optical image and by T_f on the flattened surface, as illustrated in figure 9.



(a) (b)

Figure 8: Mesh of Frog and its parametrisation.

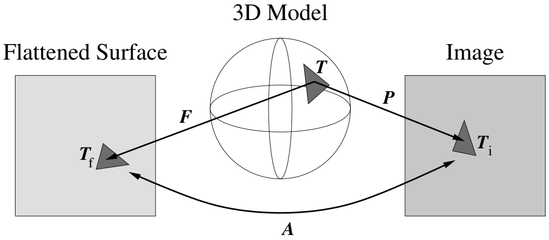


Figure 9: Relation between 3D model, optical image and flattened surface. T is a triangle, F flattening, P projective mapping, A affine mapping.

One can readily determine the affine transformation between T_i and T_f, which gives the correspondence between the points of the triangles. (Note that the affine transformation is unique for each triangle pair.) The algorithm of the conversion is the following:

```
For each triangle $T$ of 3D mesh
    If $T$ completely visible
      Projection: $T_i \leftarrow P \cdot T$.
      Flattening: $T_f \leftarrow \mathrm{FLAT}(T)$.
      Affine transformation: $A \leftarrow \mathrm{AFFINE}(T_f, T_i)$.
      For each point $\mathbf{u_f} \in T_f$:
          $\mathrm{Colour}_f(\mathbf{u_f}) \leftarrow \mathrm{Colour}_i(A \cdot \mathbf{u_f})$.
      End for.
    End if.
End for.
```

Conversion of optical images provides partial texture maps. To obtain the entire textured surface, one needs to merge these texture maps. Since flattening preserves the topology of the mesh, the texture coordinates of the partial texture maps are consistent. The only problem is that of the overlapping areas, where texture maps must be smoothly interpolated.

We have tested three methods for handling the overlapping areas. The essence of the first method is to decide for each triangle which view it is mostly visible from. The second method tries to improve the first one by blending the views. Finally, the third method applies the multiresolution spline technique [Burt and Adelson 1983] for blending the images. Using the blending methods the border between the neighbouring texture maps becomes nice and smooth, as one can see in figure 10. The difference between the results of the second and the third method is insignificant.

As already mentioned, complex meshes need to be cut into pieces before flattening. The pieces are textured separately; however, re-merging them yields seams between the borders. These artefacts can be eliminated by the alpha blending technique. This technique guarantees the continuity of the texture in the re-merged model, as illustrated in figure 11.

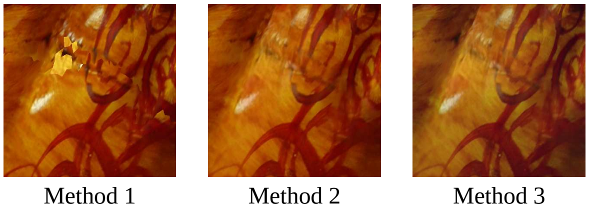

Method 1
Method 2
Method 3

Figure 10: Differences between the three merging methods.

4 Using photometric data to obtain bump maps

The final step of building a photorealistic 3D model by fusing multimodal data is adding the surface roughness in the form of a bump map that locally perturbs the normal vector of the measured smooth surface. We are currently working on this problem using the photometric stereo approach [Forsyth and Ponce 2003]. Since this work is in progress and the development of the method has not yet been finished, in this section we show just a few initial results demonstrating the feasibility of the idea.

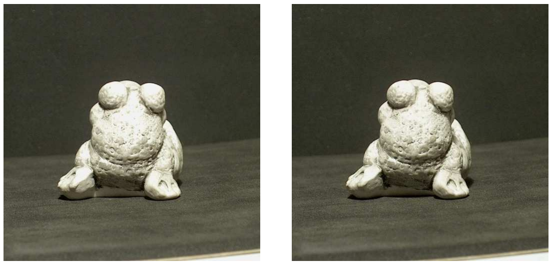


Figure 12: Two of Frog images for photometric stereo.

The traditional photometric stereo assumes a fixed camera setup with a certain number of pointwise lighting sources whose orientations with respect to the object are known. A collection of images is taken by successfuly switching on each source separately. Two images of the Frog collection for the photometric stereo are shown in figure 12. The variation of the pixel values under the varying illumination can be used to obtain the bump map of the surface, or the surface normal in the global co-ordinates.

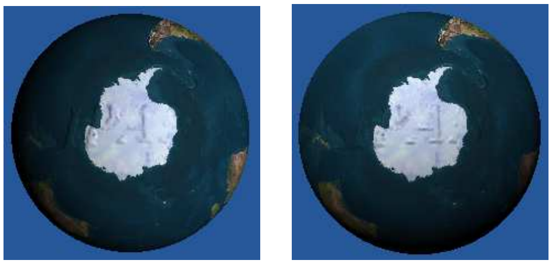


Figure 13: Two of synthetic Globe images for photometric stereo.

Modern methods for the photometric stereo [Forsyth and Ponce 2003] do not assume that the orientations of the lighting sources are known. To provide additional constraints, more images are taken, an overdetermined system is received, and a solution for the surface normal is obtained in the least squares sense. The normal is then integrated to obtain the surface, while taking into account the bas-relief ambiguity. We use a modified photometric algorithm to obtain the normal map on the flattened surface.

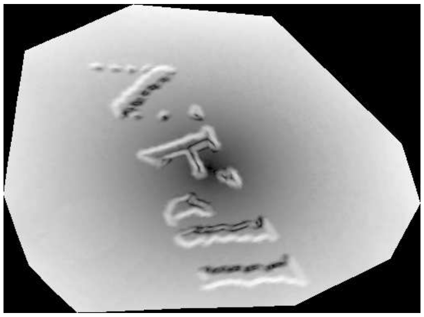


Figure 14: Normal map on the flattened Globe surface.

Figures 13 and 14 demonstrate the feasibility of our approach. Figure 13 shows two of synthetic Globe images we use to test the photometric algorithm. A synthetic globe was created and a relief inscription, IPAN, was put onto the surface. (IPAN is the abbreviation for Image and Pattern Analysis group of our lab.) A part of the word is visible in figure 13. The images simulate the intensity variation as the position of the illumination sourse changes. Figure 14 shows the obtained normal map on the flattened surface of the Globe. Most of the relief inscription has been successfully recovered.

5 Tests

Our photorealistic modelling system has been tested both on synthetic and real data. The synthetic data provides the ground truth necessary for assessing the performance of the system in terms of precision and computational efficiency. In this section, we give further examples of processing real measured data and creating high-quality models using the algorithms described above.

The already mentioned Bear dataset, as well as the Frog, the Shell, and the Cat datasets were acquired by a 3D laser scanner and a high-resolution digital camera. In each case, the complete textureless 3D model (triangular mesh) was obtained by the surface registration algorithm presented in section 2 and the triangulator [Kós 2001]. Then, some 5–6 images of each object were taken. The images were registered to the 3D model and blended by the methods presented in section 3. For the blending, the 3D models were interactively cut in half. The halves were handled separately and merged only at the end of the process. The results can be seen in figures 11, 15, 16, and 17.

6 Conclusion

We have presented a software system for building photorealistic 3D models. It operates with accurate 3D model measured by laser scanner and high quality images of the object acquired separately by a digital camera. The complete 3D model is obtained from partial surface measurements using a genetic based pre-registration algorithm followed by a precise iterative registration procedure. The images are registered to the 3D model by minimising a photo-consistency based cost function using a genetic algorithm. Since textures extracted from images can only cover parts of the 3D model, they should be merged to a complete texture map. A novel method is used to combine partial texture mappings using surface flattening. Test results with real data demonstrate the efficiency of the proposed methods. A high-quality model of a relatively small object can be obtained within two hours, including the processes of 3D scanning and photography. We are currently working on improving our method that adds surface roughness by measuring the bump maps with photometric stereo.

Acknowledgement

This work is supported by EU Network of Excellence MUSCLE (FP6-507752).

References

BERNARDINI, F. 2002. Building a digital model of Michelangelo's Florentine Pietà. *IEEE Comp. Graphics & Applications 22*, 1, 59–67.

BESL, P., AND MCKAY, N. 1992. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence 14*, 239–256.

BURT, P. J., AND ADELSON, E. H. 1983. A multiresolution spline with application to image mosaics. *ACM Trans. Graph. 2*, 4, 217–236.

CHETVERIKOV, D., STEPANOV, D., AND KRSEK, P. 2005. Robust Euclidean Alignment of 3D point sets: the Trimmed Iterative Closest Point algorithm. *Image and Vision Computing 23*, 299–309.

CHETVERIKOV, D., JANKÓ, Z., LOMONOSOV, E., AND EKÁRT, A. 2006. Creating photorealistic models by data fusion with genetic algorithms. Studies in Fuzziness and Soft Computing. Springer. In print.

CLARKSON, M. 2001. Using photo-consistency to register 2D optical images of the human face to a 3D surface model. *IEEE Tr. on PAMI 23*, 1266–1280.

DAVID, P. 2002. SoftPOSIT: Simultaneous pose and correspondence determination. In 7th *European Conf. on Computer Vision*, 698–714.

FORSYTH, D., AND PONCE, J. 2003. *Computer Vision: A Modern Approach*. Prentice Hall.

HARALICK, R. 1989. Pose estimation from corresponding point data. *IEEE Tr. on SMC 19*, 1426–1445.

HARTLEY, R., AND ZISSERMAN, A. 2000. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press.

IKEUCHI, K. 2003. The great Buddha project: Modeling cultural heritage for VR systems through observation. In *IEEE ISMAR03*.

JANKÓ, Z., AND CHETVERIKOV, D. 2004. Photo-consistency based registration of an uncalibrated image pair to a 3D surface model using genetic algorithm. In 2nd *Int. Symp. on 3D Data Processing, Visualization & Transmission*, 616–622.

KÓS, G., AND VÁRADY, T. 2003. Parameterizing complex triangular meshes. In 5th *International Conf. on Curves and Surfaces*, 265–274.

KÓS, G. 2001. An algorithm to triangulate surfaces in 3D using unorganised point clouds. *Computing Suppl. 14*, 219–232.

KUTULAKOS, K., AND SEITZ, S. 1993. *A Theory of Shape by Space Carving*. Prentice Hall.

LEVENTON, M., WELLS III, W., AND GRIMSON, W. 1997. Multiple view 2D-3D mutual information registration. In *Image Understanding Workshop*.

LOMONOSOV, E., CHETVERIKOV, D., AND EKÁRT, A. 2006. Pre-registration of arbitrarily oriented 3D surfaces using a genetic algorithm. *Pattern Recognition Letters*. In print.

M. LEVOY. 2000. The digital Michelangelo project. *ACM Computer Graphics Proceedings*, 131–144.

YEMEZ, Y., AND SCHMITT, F. 2004. 3D reconstruction of real objects with high resolution shape and texture. *Image and Vision Computing 22*, 1137–1153.

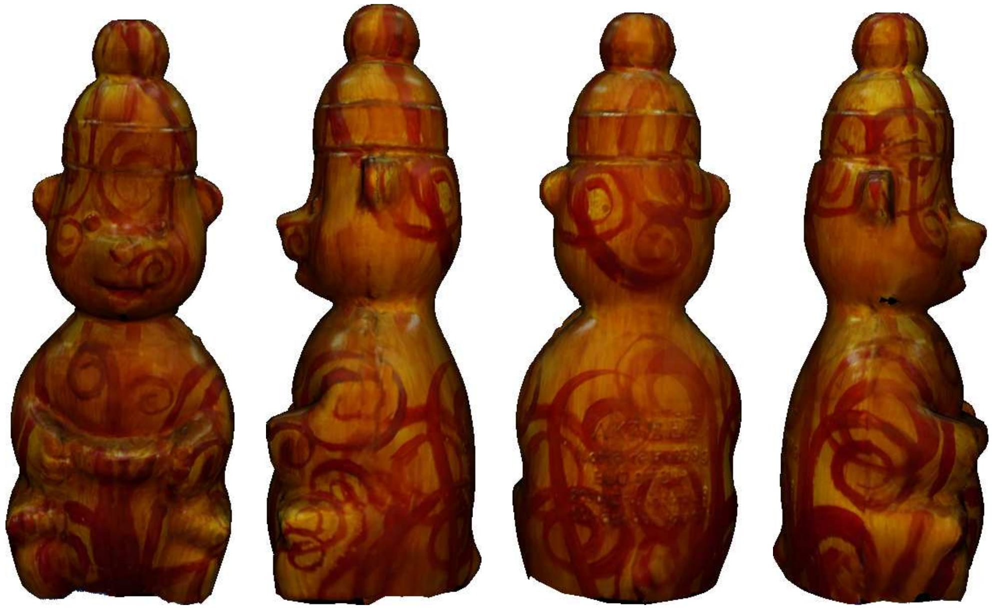
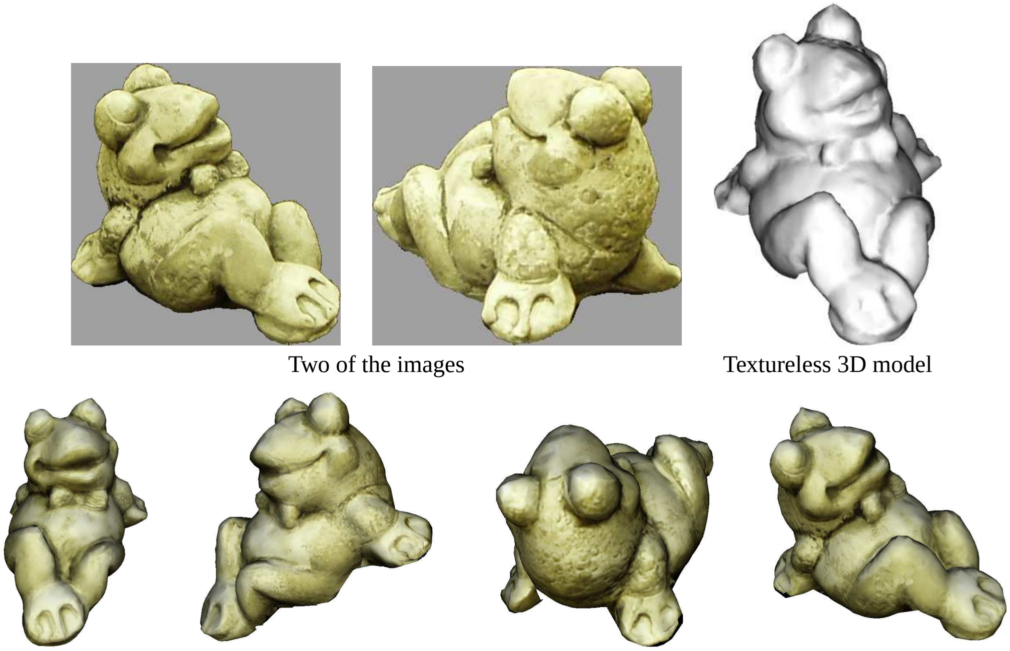


Figure 11: Result of texturing the 3D model of Bear.

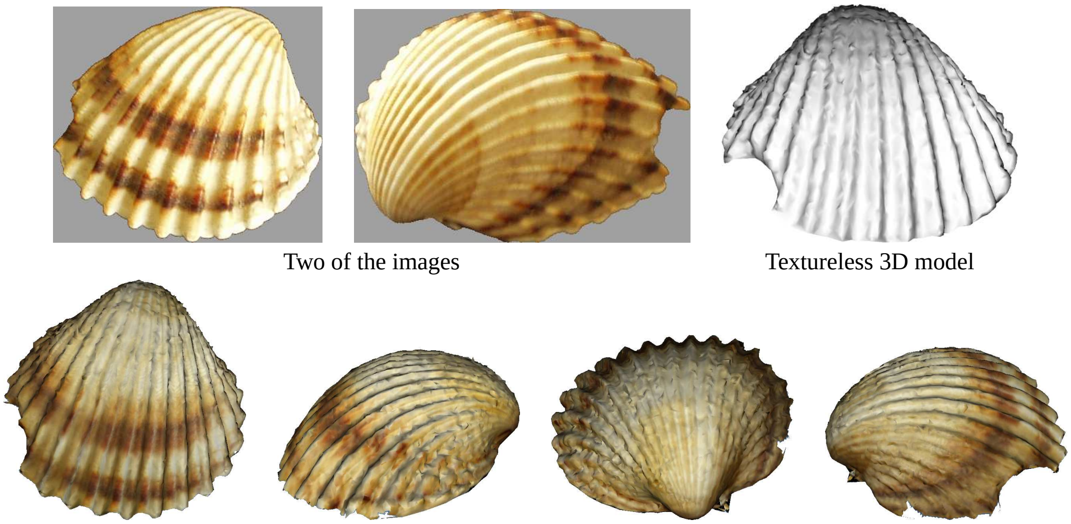


Two of the images

Textureless 3D model

Views of textured 3D model

Figure 15: Building photorealistic model of Frog.

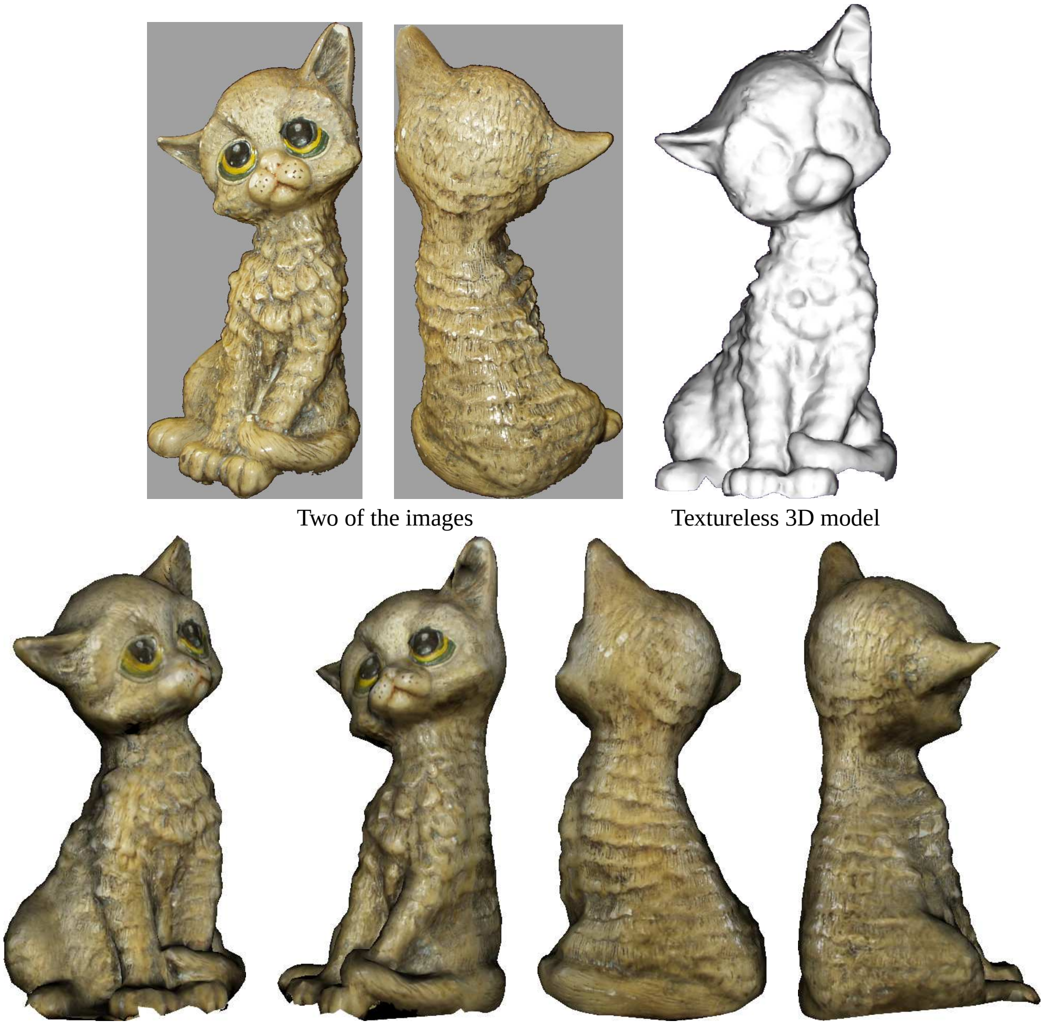


Two of the images

Textureless 3D model

Views of textured 3D model

Figure 16: Building photorealistic model of Shell.



Two of the images

Textureless 3D model

Views of textured 3D model

Figure 17: Building photorealistic model of Cat.